STUDIA INFORMATICA 2010
Volume 31 Number 2B (90)

Wojciech AMBROZIK
KAMSOFT SA
Romuald BŁASZCZYK
Uniwersytet Śląski, Instytut Informatyki

PROBLEMATYKA BEZPIECZEŃSTWA W WYBRANYCH TECHNIKACH KRYPTOGRAFICZNYCH

Streszczenie. Celem pracy jest porównanie współczesnych algorytmów faktoryzacji w kontekście kryptosystemu RSA. W tym celu przeprowadzono praktyczne badania różnych metod oraz ocenę ich efektywności. Aby realizować ten cel konieczna była gruntowna analiza tych metod, ale ograniczone możliwości pracy uniemożliwiają jej przedstawienie.

Słowa kluczowe: bezpieczeństwo, RSA, kryptoanaliza

SECURITY PROBLEM IN CHOSEN A CRYPTOGRAFIC METHODS

Summary. The aim of the paper is to compare modern algorithms of factorization in the context of RSA cryptosystem. Author presents results of practical experiments based on different methods and evaluations of their efficiency. Obtained results needed thorough analysis of formalization of these methods. Because of limitations of the paper it is not possible to present the whole analysis.

Keywords: security, RSA, cryptoanalytic

1. Wprowadzenie

Praca ta jest pewnym dopełnieniem prac z zakresu kryptoanalizy RSA. Nie pretenduje ona do kompleksowego ujęcia tej tematyki (praktycznie byłoby niemożliwe), ale porusza pewne niszowe rozwiązania w tym zakresie i dokumentuje je wynikami badań praktycznych.

Od ponad trzydziestu lat kryptosystem RSA jest głównym celem ataków kryptoanalityków, a za jego złamanie wyznaczono sowite nagrody finansowe. Jak się okazuje, RSA prze-

trwał lata intensywnej kryptoanalizy i do dnia dzisiejszego jest bezapelacyjnym liderem w „konkursie na bezpieczną kryptografię jutra". Wysoki poziom bezpieczeństwa – jaki zapewnia – przyczynił się do praktycznego wykorzystania niniejszego kryptosystemu w wielu protokołach sieciowych, jak chociażby TLS/SSL lub SSH [4, 8]. To właśnie trudność faktoryzacji dużych liczb, czyli problem rozkładu liczby na czynniki pierwsze, pozwala uznać współczesną bankowość i handel elektroniczny za bezpieczne. Jak wynika z badań nad kryptoanalizą RSA, każda inna próba złamania niniejszego kryptosystemu, również za pomocą logarytmu dyskretnego, jest przynajmniej tak skomplikowana obliczeniowo, jak zagadnienie faktoryzacji liczb całkowitych. Tak więc w pracy tej spośród całego szeregu zagadnień kryptoanalizy RSA wybrano te, które dotyczą rozkładu liczb. Istnieją różne algorytmy faktoryzacji wykorzystujące zaawansowane mechanizmy teorii liczb, jednak do dnia dzisiejszego nie znaleziono efektywnego, który złamie RSA w akceptowalnym czasie. Oczywiste jest, że odkrycie wydajnego i funkcjonalnego algorytmu rozkładu liczb na czynniki pierwsze bądź znalezienie dowolnej innej skutecznej metody złamania RSA wywrze ogromy wpływ na dzisiejsze społeczeństwo informacyjne. Przedstawione rozważania bazują na aktualnych informacjach z zakresu kryptoanalizy i dają możliwość szacowania bezpieczeństwa najpopularniejszego algorytmu kryptograficznego RSA, którego bezpieczeństwo opiera się na problemie faktoryzacji liczb całkowitych [2, 5, 10]. Niektóre algorytmy faktoryzacji są tak skonstruowane, by działały lepiej w przypadku, gdy liczba całkowita n podlegająca faktoryzacji jest szczególnej postaci; są to tzw. algorytmy faktoryzacji szczególnego przeznaczenia. Czas wykonania algorytmów ogólnego przeznaczenia zależy wyłącznie od rozmiaru n. Brak jest praktycznych badań w dostępnej literaturze oraz zbiorczego zestawienia wyników testów. Należy więc przypuszczać, że większość wniosków wysuwanych przez autorów tychże prac wynika z czysto hipotetycznych oszacowań asymptotycznych, które często znacznie odbiegają od stanu faktycznego, szczególnie dla względnie małych wartości. W związku z powyższym przedstawione w niniejszej pracy wyniki badań oparte są na empirycznych doświadczeniach wykonanych za pomocą stworzonej aplikacji. Szczególnie interesujące z punktu widzenia badań jest sprawdzenie efektywności działania algorytmów faktoryzacji ogólnego przeznaczenia, które ze względu na asymptotyczne tempo wzrostu, wykazują lepsze właściwości od algorytmów szczególnego przeznaczenia. Wiele uwagi poświęcono metodzie faktoryzacji opartej na krzywych eliptycznych [3, 6, 9], wykorzystującej operacje podwojeń i dodawań grupowych. Z uwagi na brak badań tego typu, przebadano wpływ wielkości parametru k na prawdopodobieństwo pomyślnego rozkładu na czynniki pierwsze.

2. Algorytmy faktoryzacji

Algorytmy faktoryzacji szczególnego przeznaczenia charakteryzują się jakąś własnością. Zwykle taką szczególną własnością jest to, że czynniki rozkładanej liczby są małe. Do innych

takich własności można zaliczyć fakt, że czynniki mają specjalną matematyczną postać, np. jeśli p jest liczbą pierwszą i liczba $2^p - 1$ jest złożona, to wszystkie jej dzielniki muszą przystawać do 1 modulo $2p$ ($2^{11} - 1 = 23 \cdot 89$ oraz $23 \equiv 89 \equiv 1 \pmod{22}$). Ponadto czas wykonania takich algorytmów zazwyczaj zależy od pewnych własności czynników liczby n. Przykładami algorytmów faktoryzacji [3] szczególnego przeznaczenia są:

- próbne dzielenie,
- algorytm rho Pollarda,
- algorytm $p - 1$ Pollarda,
- algorytm oparty na krzywych eliptycznych (ECM).

Określenie „ogólna metoda faktoryzacji“ oznacza, że czas faktoryzacji liczby za pomocą tej metody jest mniej więcej tego samego rzędu, co czas faktoryzacji tą metodą innej liczby (ale tej samej wielkości). Rozbicie tą metodą liczby 100-cyfrowej na iloczyn liczby jednocyfrowej i 99-cyfrowej zajmuje tyle samo czasu, co rozbicie innej liczby 100-cyfrowej na iloczyn dwóch liczb 50-cyfrowych. Trzy przedstawione w niniejszym rozdziale metody faktoryzacji nie zależą od jakiś specjalnych własności zarówno rozkładanych liczb, jak i ich czynników. Czas wykonania algorytmów ogólnego przeznaczenia zależy wyłącznie od rozmiaru n. W niniejszej pracy ujęto następujące algorytmy z tej grupy:

- metoda ułamków łańcuchowych,
- algorytm sita kwadratowego,
- algorytm ogólnego sita ciała liczbowego.

Algorytmy obydwu grup były przedmiotem badań na potrzeby niniejszej pracy.

3. Implementacja systemu

Aplikacja dokonuje rozkładu liczb złożonych na czynniki (zazwyczaj) pierwsze za pomocą wybranego przez użytkownika algorytmu. Wszelkie informacje z przebiegu algorytmu, w tym w szczególności nietrywialne dzielniki faktoryzowanej liczby, wyświetlane są użytkownikowi w oknie informacyjnym. Aplikację stworzono w środowisku NetBeans IDE 6.5.1, w związku z tym do jej uruchomienia niezbędny jest system operacyjny z zainstalowanym środowiskiem Java Runtime Environment w wersji 1.6.0. Wymagania sprzętowe aplikacji

sprowadzają sie do wymagań stawianych systemowi operacyjnemu, na którym to niniejszy program jest uruchamiany.

Użytkownik systemu może skorzystać z dwóch alternatywnych opcji: określić liczbę bitów liczb *P* oraz *Q*, co oznacza, że zezwala aplikacji na wylosowanie liczb całkowitych, których liczba bitów w wyniku przedstawienia niniejszych liczb w systemie binarnym, nie przekracza wartości podanej przez użytkownika (tak wylosowane liczby *P* oraz *Q* mnożone są przez siebie i w efekcie użytkownik uzyskuje ich iloczyn *N*) lub też – w sposób bezpośredni – określić wartość liczby *N*, przy czym *N* nie musi być iloczynem dwóch liczb pierwszych, lecz dowolną liczbą całkowitą > 2. Dane wprowadzane przez użytkownika poddawane są walidacji. W przypadku wprowadzenia błędnych danych wejściowych, aplikacja informuje użytkownika stosownym komunikatem.

W celu rozkładu *N* na czynniki należy wybrać z listy algorytm faktoryzacji (domyślnie dzielenie próbne). Niektóre algorytmy, a w tym:

- algorytm oparty na krzywych eliptycznych,
- metoda ułamków łańcuchowych,
- algorytm sita kwadratowego,

wymagają od użytkownika podania dodatkowych parametrów. Dla algorytmu opartego na krzywych eliptycznych dodatkowym parametrem jest wartość *k*. Im wartość *k* jest większa, tym większa jest szansa znalezienia czynnika pierwszego *N*, jednak czas działania algorytmu znacznie się wydłuża. W przypadku metody ułamków łańcuchowych oraz algorytmu sita kwadratowego istnieje pewna wspólna grupa parametrów. Rozmiar bazy czynników pierwszych *FB* określa liczbę kolejnych liczb pierwszych (począwszy od 2, 3, 5, 7, ...) wykorzystywanych do przesiewania. W metodzie ułamków łańcuchowych baza *FB* składa się ze wszystkich liczb pierwszych. W algorytmie sita kwadratowego baza *FB* zawiera tylko te liczby pierwsze, dla których *N* jest resztą kwadratową modulo *p* (przy czym *p* jest kolejną „małą" liczbą pierwszą). Ponadto wszystkie metody ogólnego przeznaczenia bazują na wspólnym pomyśle znajdywania kongruencji Legendre'a. W celu otrzymania kwadratów po obu stronach należy pomnożyć niektóre ze zgromadzonych kongruencji przez siebie. Obie strony kongruencji będą kwadratami o ile suma odpowiednich wektorów z wykładnikami będzie wektorem zerowym modulo 2, co w efekcie daje duże prawdopodobieństwo znalezienia nietrywialnego czynnika pierwszego *N*. Fakt ten wymusza wprowadzenie parametru określonego mianem maksymalnej liczby wymnażanych kongruencji. Kolejny parametr określa rozmiar macierzy, z której to program losuje kongruencje.

W przypadku metody ułamków łańcuchowych kongruencje powiązane są pośrednio z reduktami ułamka łańcuchowego $\sqrt{N}$. W algorytmie sita kwadratowego wynikają one z rozkładu na czynniki pierwsze wielomianów $q(x)$.

W przypadku pomyślnego wykonania algorytmu z okna informacyjnego użytkownik może uzyskać następujące informacje:

- Znalezione dzielniki liczby N. W przypadku liczb RSA dzielniki te są zawsze pierwsze, natomiast w przypadku dowolnej naturalnej liczby złożonej to, czy wszystkie znalezione dzielniki są pierwsze zależy od wybranej metody faktoryzacji. Niektóre algorytmy rozkładu liczb są tak skonstruowane, że nie nadają się do uzyskania pełnej faktoryzacji. W dalszej części rozdziału fakt ten wyjaśniono na podstawie metody ECM.
- Czas faktoryzacji liczby określony zgodnie z normą ISO 8601 (hh:mm:ss).

4. Badanie czasu faktoryzacji

Głównym celem niniejszej pracy jest przeprowadzenie badań potwierdzających nieprzełamywalność schematu szyfrowania RSA. Oczywiście, tak jak już wspomniano we wstępie pracy, badania dotyczą porównania efektywności algorytmów rozkładu liczb na czynniki pierwsze. Wykorzystując stworzoną aplikację przebadano czas faktoryzacji w kontekście wybranych metod, wielkości faktoryzowanego modułu *n*, jak również w kontekście różnych rzędów wielkości czynników pierwszych n. Ponadto jako kryterium akceptowalności algorytmów ustalono maksymalny czas rozkładu liczby na 12 godzin. Jako kolejny cel badań przyjęto próbę określenia wpływu wielkości parametru *k* na prawdopodo-bieństwo pomyślnej faktoryzacji liczb RSA stosując jako metodę rozkładu algorytm oparty na krzywych eliptycznych. Sprawdzono także efektywność algorytmów szczególnego przeznaczenia w konfrontacji z algorytmami z drugiej grupy w kontekście architektury jednokomputerowej. Większość wyników badań przedstawiono w postaci tabel i uzupełniono je o wykres. Do testów wykorzystano komputer o poniższej specyfikacji:

- System operacyjny: Windows Vista Ultimate Service Pack 1 – wersja 32-bitowa,
- Procesor: Intel Core 2 Duo E8200 (2.66 GHz @ 3.2 GHz),
- Pamięć operacyjna: 2,00 GB.

Tabela 1 zawiera zestawienie liczb RSA, na których przeprowadzono testy.

Jak widać, czas działania algorytmów szczególnego przeznaczenia w większej mierze zależy od rozmiaru najmniejszego czynnika pierwszego *N* niż od samego *N*. Fakt ten można zaobserwować analizując wyniki tabeli 2. Wzrost rzędu wielkości czynnika pierwszego *Q* przy tej samej wielkości czynnika *P* nie miał zbyt dużego wpływu na czas działania tych algorytmów. Inaczej sytuacja przedstawia się w przypadku algorytmów ogólnego przeznaczenia. Stała (mała) wartość czynnika pierwszego *P* nie ma w tym przypadku znaczenia. Czas działania tych algorytmów uzależniony jest od rozmiaru *N*, a nie od rozmiaru *P* lub *Q*. Na domiar tego powszechnie sądzi się, że algorytmy szczególnego przeznaczenia należy stoso-

wać dla względnie małych liczb złożonych (mniej niż 100 cyfr). Natomiast w przypadku liczb większych warto posłużyć się jedną z silniejszych metod faktoryzacji – ogólnego przeznaczenia. Oczywiście, analizując wyniki przedstawione w tabeli, nietrudno zauważyć przewagę algorytmów szczególnego przeznaczenia nad drugą grupą, co jest niejako sprzeczne z założeniami teoretycznymi zawartymi w literaturze.

Tabela 1

Zestawienie testowych liczb RSA

Oznaczenie	N = P * Q	P	Q
N_1	523087	691	757
N_2	811053008473	1021	794371213
N_3	670739376082428491	643	1043140553782937
N_4	956013821524748316052549	877	1090095577565277441337
N_5	725899553626667170952795940263221	853	850995959703009579077134748257
N_6	866253044330073670716930883747423	856053977	1011914047015838664478199
N_7	98901523498149201912657347 6669767	1123393562401223	880381789679744129
N_8	57761408703535983987981965 5604087	9487794586091718480 11	608796998917
N_9	74343104758255577154181767 07502539622236270273218 97	1143345898995104294 434292852930153683	650224090746259
N_{10}	107204739438411289 122207451212335232 164502437267857973 211617962773828307 666885415601924965 639559512674149642 130022594604080652 472965937601033310 476844421247969778 767329291417211300 523602792166647028 366995392568580021 201249237161940910 974729281711748790 500750718116362570 135359676844524846 389740259142508869 921	522314075680700764 061076723387704208 902861220453996435 712961361009566586 019573222988603414 075651962019455836 765078702011802765 155444357817818134 658427889753775263 779258058456401640 810272061308810675 544876379368687696 647789504075165229 212072316213616381 757831298337738637 578435252480796166 2952095789653	20524957

Wynika to z faktu, iż wszelkie zaproponowane implementacje algorytmów ogólnego przeznaczenia stworzono w kontekście architektury jednokomputerowej, co czyni algorytmy te, w takiej postaci, zupełnie niepraktycznymi. Jeśli w przypadku względnie małych liczb złożonych *N* czas działania tych algorytmów jest akceptowalny, a skuteczność niemal 100%,

to w przypadku większych wartości *N* uruchomienie algorytmu często może zakończyć się niepowodzeniem.

Tabela 2

Wyniki badań

Algorytm	Parametry algorytmu	N	Liczba bitów *N*	Liczba bitów P	Liczba bitów Q	Czas rozkładu
Sito kwadratowe	Baza FB: 1000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_1	19	10	10	00:00:00,049
Ułamki łańcuchowe	Baza FB: 1000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_1	19	10	10	00:00:00,047
Sito kwadratowe	Baza FB: 1000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_2	40	10	30	00:00:01,156
Ułamki łańcuchowe	Baza FB: 1000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_2	40	10	30	00:00:00,163
Sito kwadratowe	Baza FB: 1000 Liczba kongruencji: 4 Rozmiar macierzy: 10	N_3	60	10	50	00:01:48,857
Ułamki łańcuchowe	Baza FB: 1000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_3	60	10	50	00:00:48,992
Sito kwadratowe	Baza FB: 10000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_4	80	10	70	00:06:00,501
Ułamki łańcuchowe	Baza FB: 10000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_4	80	10	70	00:00:00,703
Sito kwadratowe	Baza FB: 10000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_5	110	10	100	00:19:58,873
Ułamki łańcuchowe	Baza FB: 10000 Liczba kongruencji: 2 Rozmiar macierzy: 10	N_5	110	10	100	>12 godzin
Dzielenie próbne		N_5	110	10	100	00:00:00,218
Pollard rho		N_5	110	10	100	00:00:00,033
Pollard p-1		N_5	110	10	100	00:00:00,221
Krzywe eliptyczne	k: 100	N_5	110	10	100	00:00:00,033
Dzielenie próbne		N_{10}	1024	25	999	02:26:58,113
Pollard rho		N_{10}	1024	25	999	00:00:00,515
Pollard p-1		N_{10}	1024	25	999	00:38:02,001

cd. tabeli2

Krzywe eliptyczne	k: 100	N_{10}	1024	25	999	00:00:00,249
Dzielenie próbne		N_6	110	30	80	03:28:29,456
Pollard rho		N_6	110	30	80	00:00:00,344
Pollard p-1		N_6	110	30	80	00:00:00,812
Krzywe eliptyczne	k: 1000	N_6	110	30	80	00:00:00,297
Dzielenie próbne		N_7	110	50	60	>12 godzin
Pollard rho		N_7	110	50	60	00:02:52,106
Pollard p-1		N_7	110	50	60	00:57:54,470
Krzywe eliptyczne	k: 120000	N_7	110	50	60	00:19:28,267
Dzielenie próbne		N_8	110	70	40	>12 godzin
Pollard rho		N_8	110	70	40	00:00:08,328
Pollard p-1		N_8	110	70	40	00:00:12,609
Krzywe eliptyczne	k: 100000	N_8	110	70	40	00:12:54,452
Pollard rho		N_9	169	120	50	00:05:29,716
Pollard p-1		N_9	169	120	50	00:19:50,788
Krzywe eliptyczne	K: 450 000	N_9	169	120	50	>12 godzin

Konsekwencją tego jest fakt, że wraz ze wzrostem *N* zwiększa się rozmiar macierzy do przesiewania, jak również liniowo maleje prawdopodobieństwo gładkości dla z góry ustalonej bazy czynników *FB*. Ponadto nie istnieje reguła, która pozwalałaby określić liczbę wymnażanych kongruencji. W związku z powyższym algorytmy ogólnego przeznaczenia (w kontekście architektury jednokomputerowej) częstokroć pomijają możliwe rozwiązanie, co ma duży wpływ na ich czas działania. Dlatego każdorazowe uruchomienie algorytmu, nawet przy tych samych parametrach, może dać różny czas efektywnej faktoryzacji liczby złożonej. Oczywiście, rozwiązaniem niniejszego problemu jest zrównoleglenie obliczeń i takie rozwiązania stosuje się we współczesnej zaawansowanej kryptoanalizie RSA. Każdy z zaimplementowanych algorytmów posiada swój zrównoleglony odpowiednik. Wszystkie węzły komputera równoległego lub każdy węzeł w sieci przesiewa nad różnym zbiorem wielomianów (metoda QS) bądź reduktów (algorytm CFRAC), a znaleziona para (*x, y*) zgłaszana jest procesorowi centralnemu. Uruchomienie takich algorytmów w powyższy sposób na 100 lub nawet 100 000 komputerów ewidentnie zwiększa efektywność, jak i skraca

czas faktoryzacji liczby takimi metodami. Warto również wspomnieć, iż pomimo powszechnie znanej fatalnej złożoności algorytmu próbnego dzielenia, metoda ta znajduje pośrednio zastosowanie w wersjach podstawowych takich metod, jak:

- Pollarda $p - 1$,
- ułamków łańcuchowych,
- sita kwadratowego.

Większość zaimplementowanych w niniejszej pracy algorytmów dotyczy ich wersji podstawowych, bez modyfikacji, gdzie próbne dzielenie wykorzystywane jest do rozkładu na czynniki pierwsze „małych liczb złożonych" z wykorzystaniem bazy czynników *FB*. Oczywiste jest, że fakt ten wpływa niekorzystnie na działanie powyższych metod, a zarazem zaimplementowanych algorytmów. W związku z powyższym, narzucające się ulepszenia wyżej wymienionych metod polegałyby na zastąpieniu algorytmu próbnego dzielenia metodą Pollarda rho, która jak wynika z przedstawionych wyników badań, cechuje się największą efektywnością. Naturalnie, zamiast Pollarda rho można wykorzystać inny algorytm, jednak należy mieć na uwadze, że nie wszystkie metody nadają się do pełnej faktoryzacji „małych liczb złożonych". Poprzez „małe liczby złożone" należy rozumieć w przypadku metody ułamków łańcuchowych wyrażenia postaci $P^2 - n \cdot Q^2$, gdzie P oraz Q stanowią redukty tego ułamka, natomiast w przypadku metody sita kwadratowego są to kolejne wielomiany $q(x)$. Przykładowo, algorytm oparty na krzywych eliptycznych już z samej definicji krzywej eliptycznej uniemożliwia dokonanie pełnej faktoryzacji dowolnej liczby całkowitej n większej od zera, gdyż w celu faktoryzacji liczby konieczne jest spełnienie warunku $NWD(6, n) = 1$. Tak więc wszystkie liczby złożone mające dzielniki pierwsze 2 oraz 3 nie dają się rozłożyć na czynniki metodą ECM.

W celu dalszej analizy przeprowadzono kolejne badania określające wpływ wielkości parametru k na czas oraz prawdopodobieństwo faktoryzacji z wykorzystaniem metody ECM. Uzyskane wyniki badań przedstawiono w tabeli 3.

Pomimo iż w tabeli 3 zawarto wyniki badań tylko dla faktoryzacji liczby N_6, to wnioski z nich płynące można uogólnić na dowolną liczbę złożoną. Jak podaje dostępna literatura oraz o czym wspomniano już wcześniej, wraz ze wzrostem rzędu wielkości parametru k wzrasta prawdopodobieństwo znalezienia dzielników pierwszych. Warto jednak ponownie przypomnieć, że wartość parametru k tak naprawdę oznacza k silnia dodawań punktów nad wylosowaną krzywą eliptyczną. Oczywiście, wzrost wartości parametru k powoduje wydłużenie czasu działania algorytmu. Należy również zauważyć, że znaczne wydłużenie czasu faktoryzacji nie rekompensuje prawdopodobieństwa uzyskania oczekiwanego wyniku. Dla parametru $k = 7\,000$ uzyskano prawdopodobieństwo rzędu 20% otrzymania pełnego rozkładu liczby N_6 przy czasie faktoryzacji niespełna 4 sekundy. Co prawda, ustawienie wartości parametru $k = 75\,000$ prawdopodobieństwo to zwiększa do 60%, jednak czas potrzebny na fak-

toryzację jest wtedy rzędu 6 minut. W związku z powyższym, nasuwa się logiczny wniosek, iż bardziej opłacalne z punktu widzenia kryptoanalityka jest kilkukrotne uruchomienie algorytmu dla różnych krzywych eliptycznych dla mniejszej wartości parametru k. Wracając do analizowanego przykładu, dopiero 90-krotne uruchomienie algorytmu dla wartości parametru $k = 7\,000$ w kontekście architektury jednokomputerowej zajmie tyle czasu, co jednokrotne uruchomienie algorytmu dla wartości parametru k rzędu 75 000. Oczywiście, kolejne uruchomienia algorytmu można zrównoleglić, co ewidentnie przyśpieszy czas rozkładu uzależniony od liczby węzłów komputera równoległego. Ponadto osiągnięcie prawdopodobieństwa rozkładu liczby na czynniki rzędu 100% jest możliwe, lecz nieopłacalne. Dla analizowanego przykładu można przypuszczać, że wartość parametru k należałoby ustawić na 450 000 co wymagałoby ponad 12 godzin obliczeń. Przeprowadzono w tym zakresie badania dla takiej wartości parametru k na dziesięciu różnych komputerach. Oczywiście, każdy z komputerów cechował się różnymi parametrami, a czas efektywnej faktoryzacji wahał się w przedziale 13 - 20 godzin. Wzrost wielkości prawdopodobieństwa udanej faktoryzacji w zależności od wielkości parametru k przedstawiono ponadto na rysunku 1.

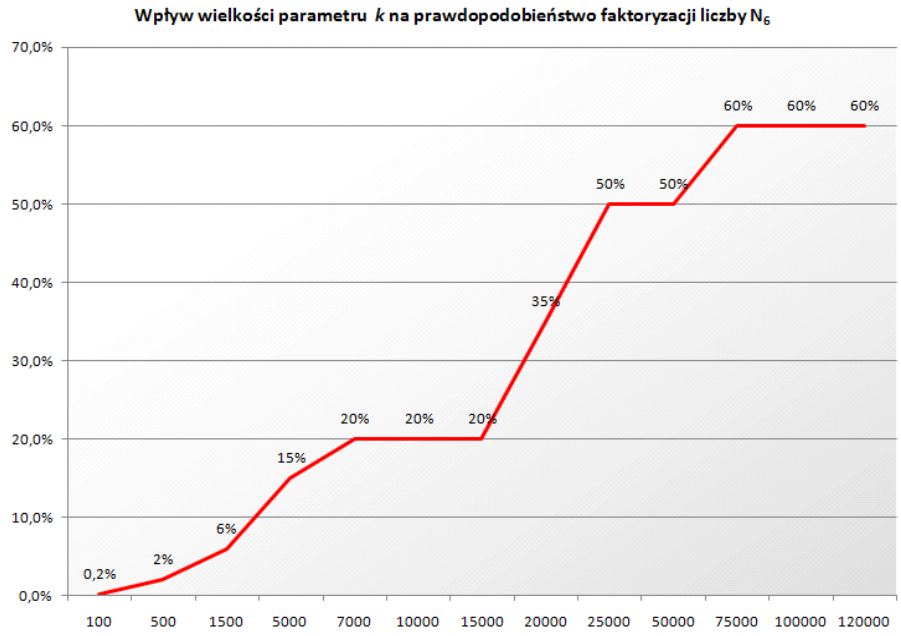
Tabela 3

Wpływ wielkości parametru k na prawdopodobieństwo faktoryzacji liczby N_{6^+}

Wartość Parametru k	Czas rozkładu	Prawdopodobieństwo rozkładu w %
100	00:00:00,057	0,2
500	00:00:00,093	2
1500	00:00:00,453	6
5000	00:00:02,256	15
7000	00:00:03,719	20
10000	00:00:06,859	20
15000	00:00:13,375	20
20000	00:00:23,860	35
25000	00:00:36,401	50
50000	00:02:19,169	50
75000	00:05:41,741	60
100000	00:12:14,490	60
120000	00:16:56,618	60

Kolejny wniosek wynika bezpośrednio z analizy wyników badań faktoryzacji liczb zamieszczonych w tabeli 2 z wykorzystaniem algorytmu opartego na krzywych eliptycznych. Jak już nieraz wspominano, wraz ze wzrostem rzędu wielkości N konieczne jest zwiększenie wartości parametru k. Jak wynika z badań, w celu faktoryzacji liczby RSA-169 niezbędne jest ustawienie wartości parametru k na większa niż 450 000. Nie ulega wątpliwości, że przy tak ustawionym parametrze algorytm wykonuje 450 000 silnia dodawań punktów nad krzywą eliptyczną E. Ponadto zastosowanie zaprezentowanego w niniejszej pracy algorytmu szybkiego obliczania kP na niewiele się zdaje, dla liczb tego rzędu wielkości. Oczywiście, można

próbować uruchomić algorytm z inną krzywą eliptyczną bądź go zrównoleglić, co wydaje się najrozsądniejszym rozwiązaniem. Zrównoleglony odpowiednik algorytmu dodawania punktów nad krzywą eliptyczną zawarł Song Y. Yan w publikacjach[10, 9].



Rys. 1. Wpływ wielkości parametru *k* na prawdopodobieństwo faktoryzacji liczby N_6
Fig. 1. Influence of largeness of parameter k on probability of factoring of number N_6

5. Zakończenie

Zgodnie z celem niniejszej pracy, dokonano porównania wybranych, współczesnych algorytmów rozkładu liczb na czynniki pierwsze. W celu oceny efektywności algorytmów wykorzystano stworzoną aplikację testując czas faktoryzacji w kontekście wybranych metod, wielkości modułu *n* oraz w zależności od różnych rzędów wielkości czynników pierwszych *n*. W eksperymentach przeprowadzonych na potrzeby niniejszej pracy, z ograniczeń czysto racjonalnych jako kryterium akceptowalności przyjęto czas rzędu 12 godzin. Porównano algorytmy ogólnego oraz szczególnego przeznaczenia. Warto przy tym zaznaczyć, że w przypadku algorytmów ogólnego przeznaczenia w zestawieniu wyników badań nie uwzględniono metody sita ciała liczbowego (NFS). Z uwagi na bardzo skomplikowane szczegóły algorytmu metody tej nie zaimplementowano. Można przypuszczać, że metoda ta będąc w grupie algorytmów ogólnego przeznaczenia, również będzie mało skuteczna w przypadku architektury jednokomputerowej. Teoretycznie efektywniejsze algorytmy ogólnego przeznaczenia okazały się gorsze od algorytmów szczególnego przeznaczenia. Wynikło to z faktu implementacji niniejszych metod w kontekście architektury jednokomputerowej.

W pracy podjęto też próby pewnych modyfikacji algorytmów Pollarda p-1, ułamków łańcuchowych oraz sita kwadratowego. Podkreślenia wymaga wykonanie testów wpływu wielkości parametru k metody ECM na czas oraz prawdopodobieństwo rozkładu. Na podstawie przeprowadzonych badań wyciągnięto wniosek, że wzrost wielkości parametru k, a jednocześnie czasu faktoryzacji jest niewspółmierny do wzrostu prawdopodobieństwa rozkładu.

Efektem przeprowadzonych badań są wyniki przedstawione w tabelach oraz na wykresie. Analizując wyniki badań trudno jednoznacznie określić, która metoda jest najlepsza, jednak potwierdza się reguła, że najpierw warto użyć algorytmów szczególnego przeznaczenia z nadzieją na znalezienie małego czynnika pierwszego p lub q, a następnie użyć algorytmów z drugiej grupy. Oczywiście, wraz ze wzrostem modułu n efektywność pozornie dobrych algorytmów szczególnego przeznaczenia maleje. W praktyce pojawienie się małego czynnika pierwszego liczby RSA nie powinno wystąpić i może jedynie wynikać z błędnej implementacji. Ponadto dla 1024-bitowych liczb RSA algorytmy ogólnego przeznaczenia również okazują się nieskuteczne, a koszty łamania niniejszego kryptosystemu w czasie jednego roku są kolosalne. W związku z powyższym trudno podważyć wysoki poziom bezpieczeństwa zapewniany przez RSA – oczywiście do chwili, w której zostanie odkryty nowy, efektywny, być może o złożoności wielomianowej algorytm rozkładu liczb na czynniki pierwsze.

Pewne obawy może budzić metoda Shora, jednakże faktoryzacja kwantowa jest wciąż w bardzo wczesnym etapie rozwoju i nie zagraża bezpieczeństwu RSA przynajmniej w chwili obecnej. Aktualny stan wiedzy pozwala jedynie na rozkładanie dwucyfrowych liczb złożonych, co czyni kwantową implementację algorytmu Shora zupełnie niepraktyczną. W realizacji tego algorytmu na komputerze „klasycznym" praktyczne wyniki są niezadowalające [11]. Ponadto, algorytm Shora w rzeczywistości nie służy do faktoryzacji, ale do szukania rzędów elementów x modulo n, co czyni go niepraktycznym aż w połowie przypadków. Jedyne rozsądne zagrożenie dla kryptosystemu RSA może stanowić kwantowy odpowiednik współczesnych algorytmów faktoryzacji, który na dzisiaj nie istnieje.

Niewątpliwie niniejsza praca nie wyczerpuje obszernej tematyki bezpieczeństwa, i w zasadzie sprowadza się do problemów związanych z bezpieczeństwem powszechnie wykorzystywanego kryptosystemu RSA. W związku z tym, w pracy tej nie poruszono metod obliczania logarytmu dyskretnego niezbędnych do ataku na kryptosystem RSA poprzez powyższy logarytm. Pominięto skomplikowane podstawy mechaniki i kryptografii kwantowej, kryptografii opartej na krzywych eliptycznych i hipereliptycznych. Warto również wspomnieć, że w ostatnim okresie czasu trwają badania nad całkowicie bezpieczną metodą szyfrowania – kryptografią DNA, która do zapisu informacji wykorzystuje kwas dezoksyrybonukleinowy, zbudowany z sekwencji czterech nukleotydów. Współczesna inżynieria molekularna pozwala manipulować ich ułożeniem. Spreparowane nici kwasu DNA mogą być nośni-

kiem zaszyfrowanej informacji, bo ciągi nukleotydów tworzących DNA można wykorzystać do zapisu informacji. W DNA można bowiem zapisać bardzo dużo danych. Gram kwasu zawiera 10^{21} nukleotydów, zatem można w nim zakodować 100 milionów terabajtów danych. Oznacza to, że w kilku gramach DNA można zmieścić wszystkie dane zmagazynowane w komputerach na całej Ziemi. Wielka pojemność nośnika pozwala na tworzenie bardzo długich kluczy kryptograficznych niedostępnych w zwykłych komputerach. Dlatego zgodnie z teorią Ashish Gehani z Duke University, zajmującego się teorią DNA, szyfru tego praktycznie nie będzie można złamać [1].

Oczywiste jest, że wyżej wymienione aspekty bezpieczeństwa pominięto z uwagi na ograniczoną objętość pracy i powinny one stać się celem dalszych badań.

BIBLIOGRAFIA

1. Gehani A., LaBean T.H., Reif J.H.: DNA – based cryptography. Springer, 2004.
2. Blake J.: Krzywe eliptyczne w kryptografii. WNT, Warszawa 2004.
3. Hoffstein J., Pipher J., Silverman J.H.: An introduction to mathematical cryptography. Springer, 2009.
4. Menezes A.J., Oorschot P.C., Vanstone S.A.: Kryptografia stosowana. WNT, Warszawa 2005.
5. Childs L.N.: A concrete introduction to higher algebra. Springer, 2009.
6. Koblitz N.: Algebraiczne aspekty kryptografii. WNT, Warszawa 2000.
7. Koblitz N.:Wykład Wykład z teorii liczb i kryptografii. WNT, Warszawa 2006.
8. Stinson D.R.: Kryptografia w teorii i w praktyce. WNT, Warszawa 2005.
9. Yan S. Y.: Teoria liczb w informatyce. WNT, Warszawa 2006.
10. Yan S.Y.: Cryptoanalytic attacks on RSA. Springer, 2008.
11. Grzywak A., Klamka J., Kapczyński A., Sobota M.: Współczesne problemy bezpieczeństwa informacji. WSB, 2008.

Recenzenci: Dr inż. Dariusz Rafał Augustyn
Prof. dr hab. inż. Andrzej Grzywak

Wpłynęło do Redakcji 31 stycznia 2010 r.

Abstract

This is an experimental work. It takes question of cryptoanalysis of method RSA. Safety assured is researched in this work by method from the point of view of its different method of cryptoanalysis practically RSA. Created software allows to compare the methods:

- trial division,
- algorithm Continued Fraction representation CFRAC,
- Pollard's $p - 1$ method,
- Pollard's ρ method,
- Quadratic Sieve Attack QS,
- Number Field Sieve Attack: GNFS and SNFS,
- Elliptic Curve Method (ECM).

Software has been made for this purpose, which enables effecting of such research. It accept certain limitations to its realization, they do not change which generality of consideration. It take certain complementary consideration also here, e.g. NP problem.

Adresy

Wojciech AMBROZIK: KAMSOFT S.A., ul. 1-Maja 133, 40-235 Katowice, Polska.
Romuald BŁASZCZYK: Uniwersytet Śląski, Instytut Informatyki, ul. Będzińska 39, 41-200 Sosnowiec, Polska, romuald.blaszczyk@us.edu.pl